Thoughts on Slow Controls

Nova Electronics Meeting
Fermilab
9-10 May 2006

Craig Dukes and Andrew Norman

UVa Group

- Senior members
  - Craig Dukes
  - Ken Nelson
- Postdocs
  - Marc Buehler
  - Andrew Norman
- Discussions with Engineering School in progress
- We have one “senior” FTE available immediately, plus one graduate student

Current Activities

- D0 experiment
  - CTT and Level 2 triggers
  - Analysis
- MIPP
  - Low level
- HyperCP
  - Low level (Dukes, cospokesperson)

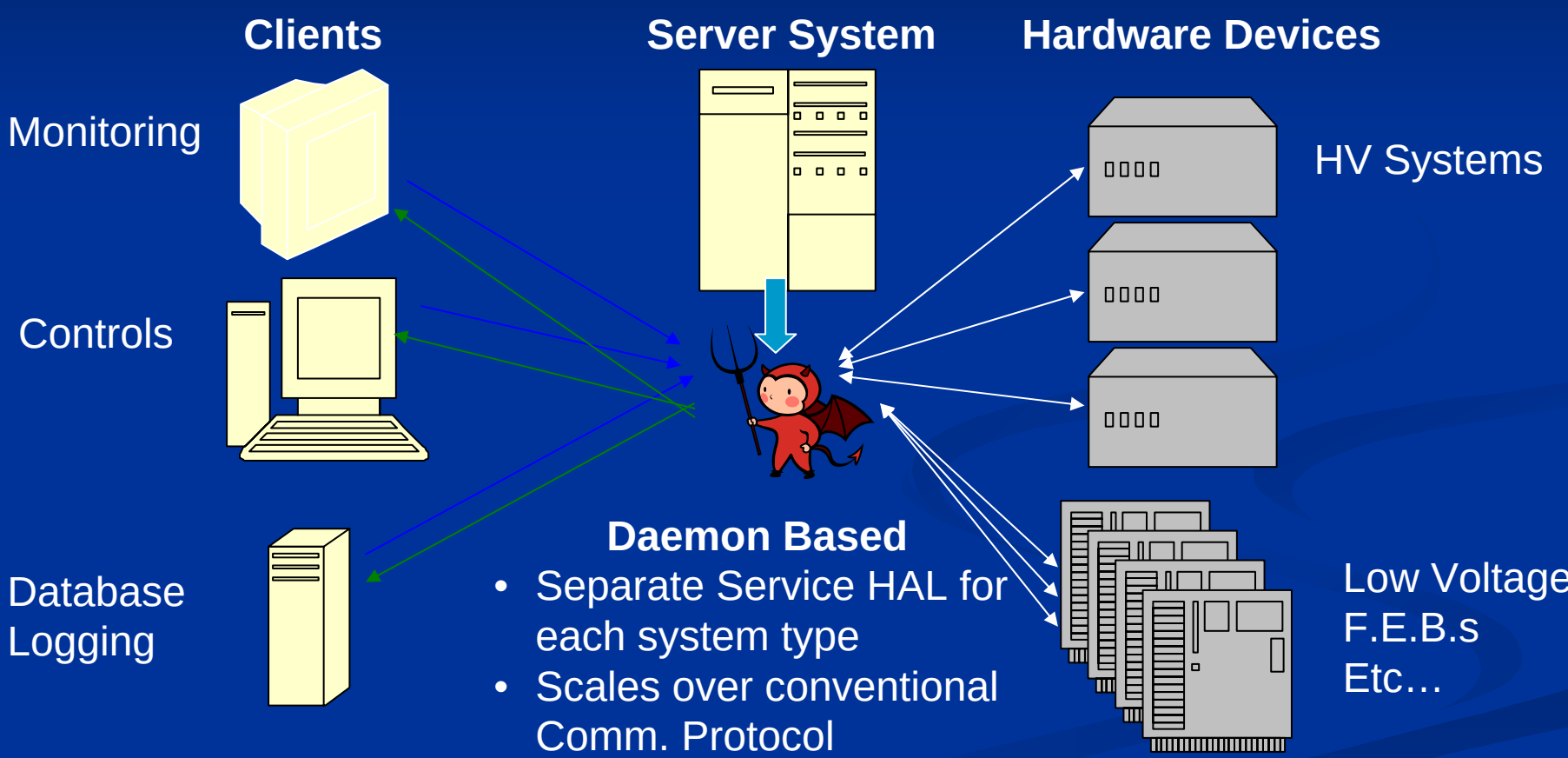
Questions

- Scope?
  - What is the line between “experiment” and “building”?
  - What is being controlled, monitored and logged?
  - Do we interface to FEBs? How?
  - Are calibration constants, pedestals, etc. included?
- Who is responsible for safety?
- Is local control (i.e., at the modules) needed? How, laptop? Presume wireless ethernet available.
- Remote monitoring from off site?
- How do we associate slow-control data with event-by-event data? Spill #, time?
- What time scale do we wish for the data logging?
- Do we want rack monitoring?
- How “smart” do we want the power distribution boxes?
  - Is that where we want to turn off individual FEBs?
  - How do we want to turn individual channels off?
- What different requirements are needed, if any, for the near detector, prototype?
- What sort of power-on requirements are there?
  - Ramp-rate limitations?
- What procedures are needed for a power failure?

Observations

- Software: use commercial off-the-shelf solutions as much as possible
  - Options: iFix, LabView, ROOT, Oracle (?), etc.
  - Avoid proprietary device-dependent solutions
- Communication: rely as much as possible on Ethernet: robust, inexpensive (finally), long-ranged
- Should be lean
- Should be robust
- Lifetime: assume at least 10 years
- Modular: must be expandable – experiment will run while being installed
- Timescale: must be available at beginning of installation, not end
- Versatile: must be able to handle production and commissioning simultaneously

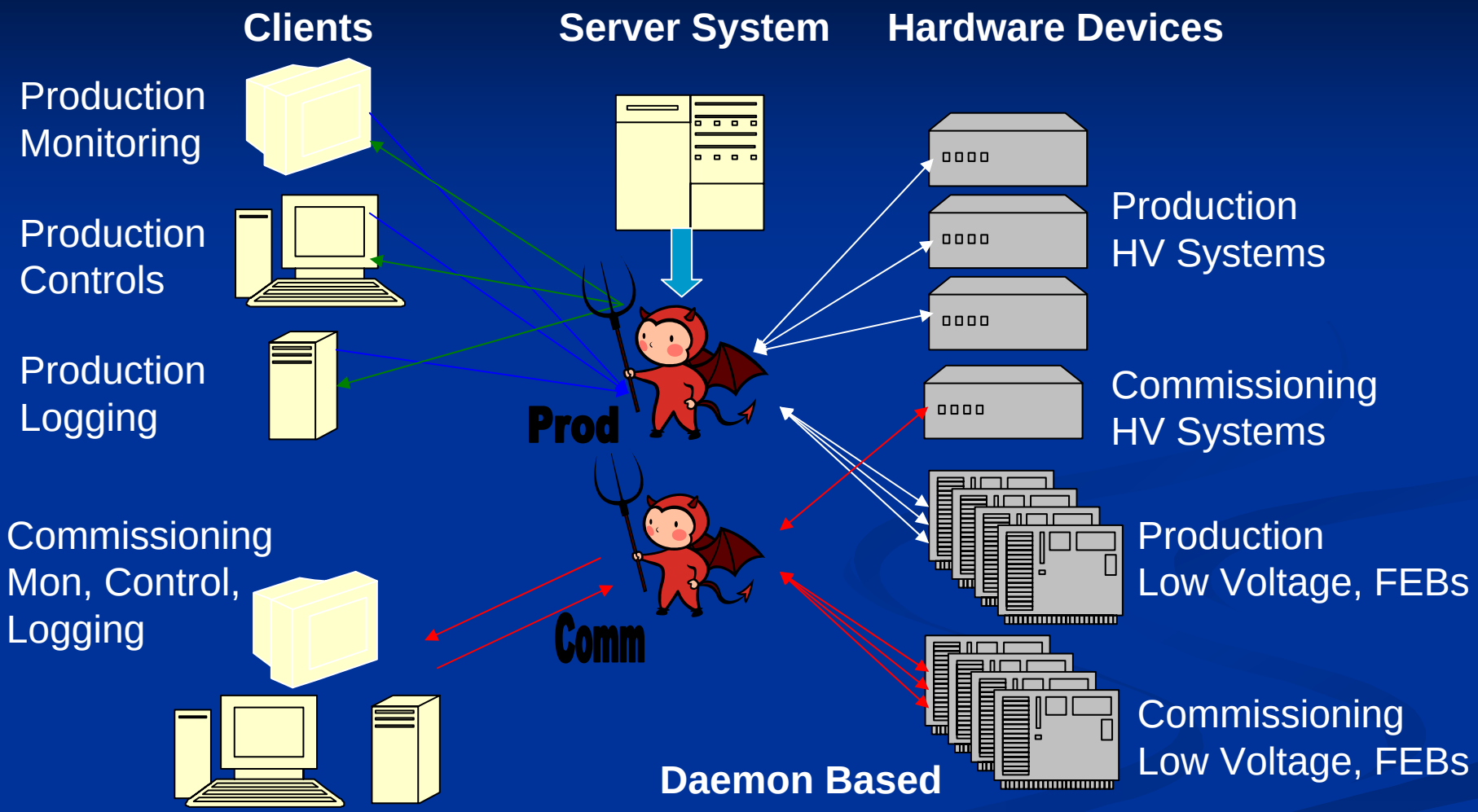
Client/Server



Daemon Based

- Separate Service HAL for each system type
- Scales over conventional Comm. Protocol
- Common Clients connect using standardize interfaces

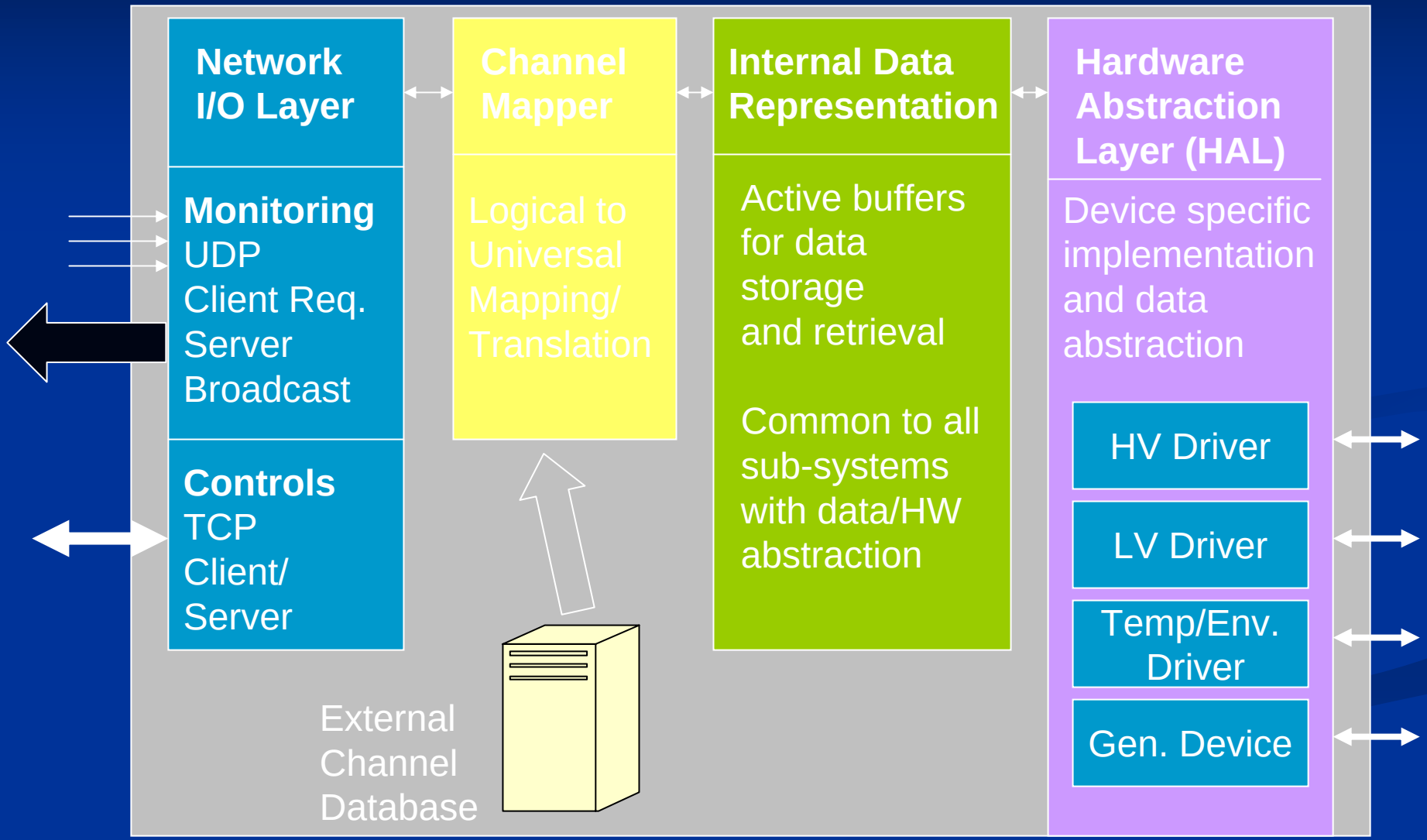
Production & Commisioning



Daemon Based

- Separate copies of the daemons run simultaneously with different hardware lists for Production and Commissioning
- Allows for seamless transitions between detector changes
- Common Clients can be used for both tasks

Daemon Structure



Example HAL Structure

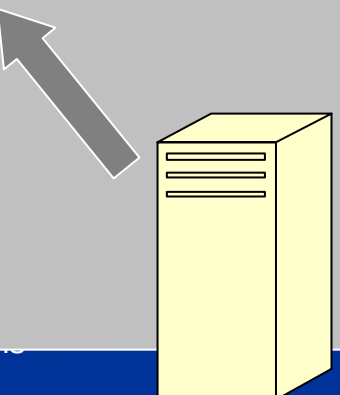
HAL Control
Dev. Initialization
Dev. ID String
Dev. Registration List
Avail. Function Mask
Aux. Function List

Address Translation
Universal Query
Address Query

Function Stack
Device ON
Device OFF
Channel ON
Channel OFF
Channel GET
Channel SET
Alarm Check
Alarm Set
Etc…….

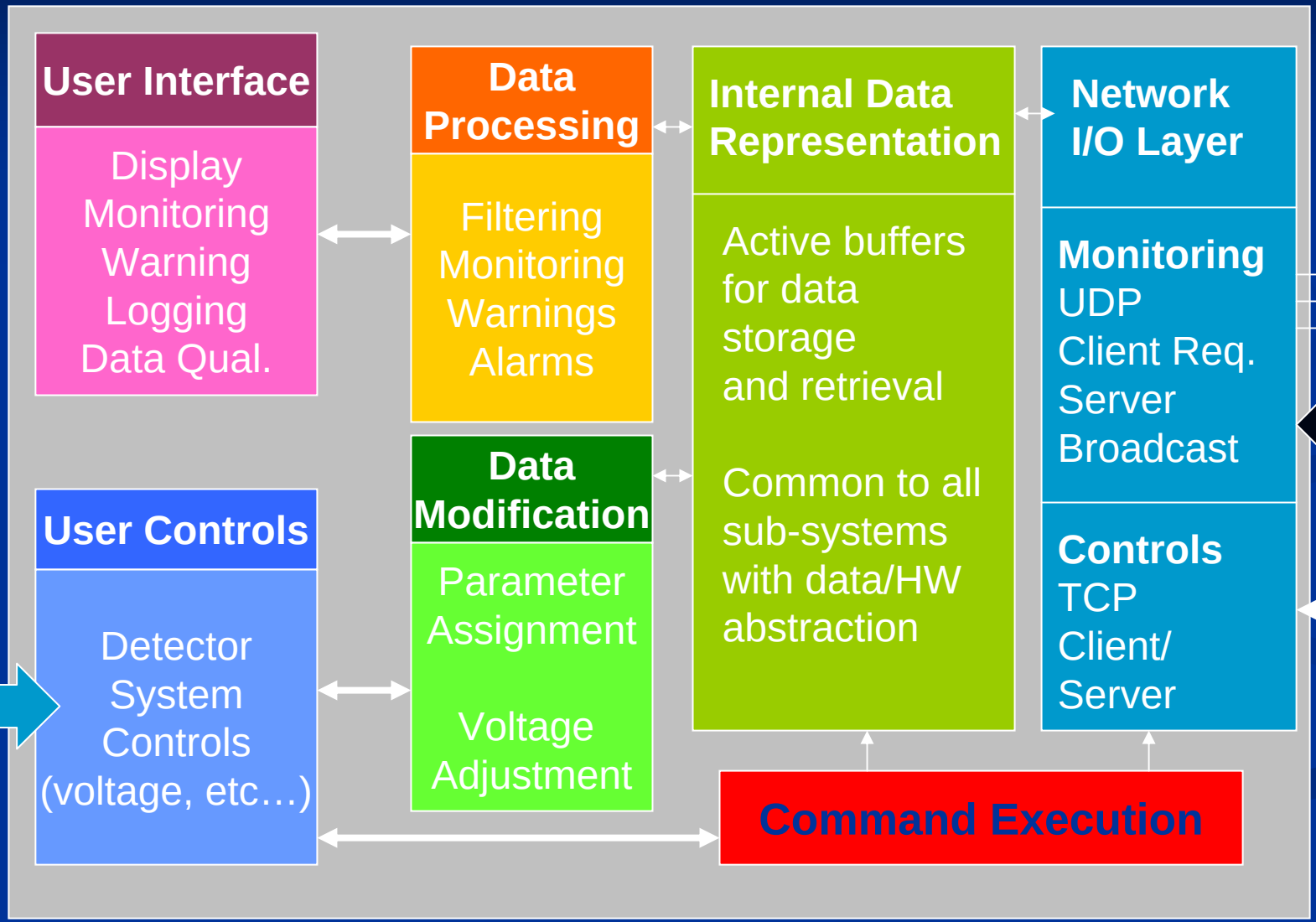
Hardware Address Map
Universal Chan Hardware Address
Universal Chan Hardware Address
Universal Chan Hardware Address

Data Stack
Device Data Value Data Value Type
Device Data Value Data Value Type
Device Data Value Data Value Type
Device Data Value Data Value Type



External Database

Client Structure



Network I/O Requirements

- Non-blocking
  - No client should be able to block/lock the daemon process preventing monitoring reads
  - Use async operation for long delay transactions (i.e. database logging)
  - Use broadcast/multicast to registered client list for monitoring/reporting instead of dedicated single pipe transactions
- Push/Pull Models
  - Client Pull. Client initiates a data pull request by establishing a dedicated pipe/socket to the server. Server responds with dedicated transmission/ verification of data operation
    - Needed for control circuits to guarantee set operations
    - Requires blocking I/O
    - High overhead, does not scale with large client numbers
  - Server Push. Client registers with server, then receives server initiated broadcasts from the server on the server's poll/report cycles
    - Non-blocking on server side
    - Low overhead, scalable to large client numbers
    - No guarantee of data reception at client
    - No retransmission on data corruption